A Proposal for Reference Architecture for Personal Assistant Software Based on SOA

S. P. Zambiasi and R. J. Rabelo

***Abstract*— Several efforts have been made towards creating personal assistant software (PAS) to help people in their daily life activities, at home or at work. In spite of the complexity PASs can have, most of the works focuses on developing solutions only for very specific tasks, without supporting integration and interoperation with other systems (including other PASs) and with enterprise systems and their respective business processes. Aiming at coping with these requirements, this work presents an open reference architecture for PASs, allowing that interoperable instances can be derived, personalized, implemented and deployed according to users' profiles and to enterprises' processes. An instance of a PAS has been derived and its results are discussed.**

***Keywords*— Personal Assistance, Reference Architecture, Interoperability, Service-Oriented Architecture.**

I. INTRODUÇÃO

A IDÉIA de softwares que dão assistência às pessoas em suas atividades não é novo e já vem sendo apresentada na forma de softwares que funcionam tal como um "secretário" [13]. Esse Software Assistente Pessoal (SAP), para [14], é um programa autônomo que oferece auxílio no gerenciamento das atividades dos seus usuários, inclusive no caso de atividades conflitantes. Segundo [12], um SAP deve automatizar tarefas rotineiras dos usuários com o intuito de liberar as pessoas para as tarefas mais importantes. Estes não podem ser visto apenas como um programa de computador. Devem ser baseados em rede, interativo, adaptativo, de propósito geral, de execução autônoma e deve poder interagir com pessoas, outros assistentes ou sistemas.

Diversas pesquisas sobre SAP têm sido efetuadas no sentido de contribuírem para os inúmeros problemas relacionados com o seu projeto e implementação. Entretanto, com base em revisão bibliográfica efetuada, observa-se que as propostas existentes focam pontos isolados do problema ou foram projetadas para atender a objetivos (i.e. tipo de tarefas) bem específicos, sem uma visão integrada e interoperável sobre as várias fontes de informação e de atividades que um usuário normalmente está envolto. Em geral, nesses SAPs compete ao usuário analisar o resultado e atuar em outra aplicação e ambiente para dar prosseguimento às suas atividades.

Também foi verificado que os SAPs não tem sido desenvolvidos para serem integrados aos processos de negócios da empresa. Na prática, isso faz com que o usuário final tenha que migrar de ambientes ao longo do seu trabalho. Por processos de negócios empresariais quer-se dizer ações e transações usuais em empresas, como compras, vendas, negociações, consulta de preços, resposta a cotações, distribuição, gestão da produção, etc. Apesar de ainda a maioria das empresas utilizar suas próprias representações de processos, observa-se o uso crescente de padrões de processos negócios empresariais, tais como UBL [21], ebXML [6] e RosettaNet [16].

A solução para os problemas relacionados com os vários aspectos de um SAP adaptativo, flexível, interoperável, interagente e integrado aos ambientes empresariais é das mais complexas e, sob certas perspectivas, ainda não plenamente resolvível com o estado das tecnologias atuais e abordagens conceituais. Neste sentido, este artigo visa trazer uma contribuição conceitual e de modelo de implementação que vai na direção daqueles requisitos. Trata-se de uma pesquisa aplicada, parcialmente exploratória e qualitativa, que visa investigar a possibilidade e importância de existir uma arquitetura de referência para SAPs que possa servir de base para instanciações particulares, abertas, baseada em padrões, flexíveis e passíveis de serem integradas a ambientes e processos de negócios de empresas.

Mais concretamente, este artigo visa apresentar uma proposta de um modelo e uma arquitetura de referência para SAPs, baseados no paradigma da arquitetura orientada a serviços, e uma instância de SAP derivada destes como resultado, ou seja, um SAP que atende em boa medida àqueles requisitos supramencionados.

Este artigo é organizado da seguinte forma. O capítulo I introduz e contextualiza o problema e o trabalho de pesquisa. O capítulo II apresenta uma definição geral de Softwares Assistentes Pessoais, requisitos básicos e uma revisão da literatura sobre estes. No capítulo III é apresentada a proposta de modelo e arquitetura de referência para SAPs. No capítulo IV há a verificação da proposta por meio de testes em uma instância implementada com base no modelo e arquitetura propostos. No capítulo V são apresentadas as considerações finais do trabalho.

II. SOFTWARES ASSISTENTES PESSOAIS

Uma vez apresentada uma conceituação bastante geral sobre a problemática e objetivos deste trabalho, este capítulo lista agora os requisitos funcionais comuns às várias definições de SAP, delineia os requisitos gerais de arquitetura e faz uma breve revisão da literatura sobre alguns dos trabalhos considerados mais relevantes ao contexto deste.

A. Definições de Softwares Assistentes Pessoais

Considerando que é muito difícil buscar uma definição que contemple as várias visões e ao mesmo tempo seja útil de ser utilizada na prática, preferiu-se descrever a definição de um SAP em termos de requisitos funcionais desejáveis, identificados por vários autores. São eles os mais comuns:

S. P. Zambiasi, Universidade Federal de Santa Catarina (UFSC), Florianópolis, Santa Catarina, Brasil, saulopz@gmail.com

R. J. Rabelo, Universidade Federal de Santa Catarina (UFSC), Florianópolis, Santa Catarina, Brasil, rabelo@das.ufsc.br

- Atuar com certa autonomia em suas tarefas para poder responder apropriadamente a cada situação [2], [7], [18];
- Ser flexível em termos de poder atuar diante de novas situações e cenários de negócios [7];
- Ser adaptável ao usuário, suas informações, preferências, necessidades, levando em conta a evolução dessas em relação ao tempo [7], [18];
- Ser interagente com seu usuário de forma a tornar possível a assistência a ele [7], [3], [18];
- Ser baseado em rede para poder buscar informações via internet e interagir com outros sistemas [2], [8];
- Ser de propósito geral, ou seja, não específico à apenas uma atividade ou um grupo de usuários [8];
- Ser adaptável de contexto, percebendo o ambiente, racionando e atuando conforme o contexto [3], [7];
- Ser integrável e interoperável aos processos de negócios da empresa [7], [22].

B. Requisitos Gerais Arquiteturais

Vários são os aspectos que devem ser considerados no projeto conceitual e de implementação de um SAP, que o fazem ser extremamente complexo de ser atacado em sua plenitude. Pode-se analisá-los sob quatro planos ou dimensões.

Num primeiro plano, há o gerenciamento do seu ciclo de vida, basicamente composto por: i) seu projeto/configuração e lançamento no sistema; ii) sua operação (i.e. a gestão das suas ações e informações durante a execução das suas tarefas); iii) sua modificação uma vez posto em operação; e iv) sua retirada do sistema (por decisão própria ou por desejo do usuário). Dentro de cada uma dessas fases há que se prever um grande conjunto de funcionalidades de suporte [22].

Num segundo plano, funcional, há que se considerar: i) como especificar as tarefas do e para o SAP; ii) como planejar as ações de cada tarefa, como supervisionar suas execuções e como atuar no caso de desvios; iii) como representar as informações e o conhecimento prévios e adquiridos do SAP ao longo do seu ciclo de vida; iv) como aprender e se adaptar em função disso; v) como monitorar sua própria existência no sistema; vi) como interpretar o desejo do usuário e convertê-lo em ações executáveis computacionalmente; vii) como fazer o SAP interoperar com os vários atores do ambiente assim como garantir uma adequada coordenação das suas ações.

Num terceiro plano, relacionado ao contexto de ação de um SAP [3]: i) contexto pessoal (preferências do usuário, históricos, e características do dispositivo); ii) contexto físico (ou de rede), considerando a representação da rede e das características de conectividade; iii) contexto de serviço, que avalia a disponibilidade de aplicações e serviços remotos; iv) contexto social, que define os diferentes tipos de papéis, regras e tarefas a serem seguidas/executadas pelo usuário em conformidade com cada contato social em cada meio e escopo de processo de negócios; v) contexto de ambiente, que gerencia as informações locais, como temperatura, tempo, previsão do tempo, localização, qualidade da rede e disponibilidade ou presença do usuário no local no momento; vi) contexto da aplicação, que avalia a compatibilidade das aplicações a serem executadas pelo assistente pessoal; vii) contexto do dispositivo, que avalia as capacidades gerais de processamento e armazenamento do dispositivo computacional onde o SAP está no momento lançado; contexto semântico, que enquadra e define a terminologia, os seus significados e correlação de conceitos e sinônimos conforme os atores com os quais o SAP se comunica.

Finalmente, um quarto plano, que envolve os aspectos de projeto de implementação e seleção das tecnologias de informação e comunicação que devem ser usadas na implementação do SAP [22]: i) software; ii) hardware; iii) *middleware*; iv) mecanismos de segurança.

C. Revisão da Literatura

Na parte da implementação, tanto iniciativas privadas como iniciativas de sistemas *open source* têm tido influência e destaque nos vários trabalhos sobre SAPs. A seguir são citados e brevemente analisados os principais trabalhos correlatos a presente proposta.

O SAP da Shelltoys, por exemplo, é um sistema para auxílio em tarefas do usuário. Contudo, ele é proprietário e fechado, possui quase nenhuma autonomia, tem pouca flexibilidade, não está preparado para se integrar a processos de negócio empresariais e se limita a gerenciar as tarefas do usuário via uma agenda de compromissos. Possui uma lista de tarefas e gera lembretes de compromissos para aniversários, atividades, reuniões e outras pequenas ações [17].

No mesmo mesmo plano, o projeto Sandy [11] foi desenvolvido para auxiliar o usuário a lembrar de assuntos (como compromissos, listas, pendências, contatos, projetos) e trabalha com troca de mensagens via SMS, Twitter e e-mail. Para o usuário utilizar o Sandy, basta enviar mensagens do tipo "Lembrar de pegar meu carro em 15 minutos", "O telefone do meu pai é 9999-9999", "lembrar de comprar mantimentos ovos", etc. O Sandy é também um projeto proprietário, fechado e foi descontinuado em 2008, conforme informado no próprio site da empresa. A grande vantagem do Sandy, e utilizada como inspiração para esta proposta, é a dinamicidade do Sandy quanto aos meios de comunicação utilizados para a interação entre o SAP e o usuário, por meio de SMS, Twitter, e-mail e outros, não engessando o usuário a apenas receber avisos por meios próprios da empresa ou por apenas um tipo de meio.

O projeto Narval (*Network Assistant Reasoning with a Validating Agent Language*, sob licença GNU LGPL) pode ser executado no próprio computador pessoal ou em um servidor remoto, e se comunica por meios normais, como e-mail, web, telnet, celular, GUI específica, etc. O sistema executa sequências de ações descritas pelo usuário para uma ampla gama de tarefas, como filtrar notícias para o jornal da manhã, ajudar a navegar na web por meio de filtro de anúncios e lixo eletrônico, cuidar de tarefas repetitivas (como responder e-mails, negociar data e hora de reuniões, etc). A sua arquitetura é bastante complexa, fazendo uso de Inteligência Artificial, Agentes, Redes de Petri, sistemas baseados em regras, programação de contratos, planejamento e aprendizagem automática. Podem ser desenvolvidos *plugins* em Python [4], [20]. Tal projeto é interessante por possuir a possibilidade de se trabalhar com comportamentos configuráveis, como *plugins*. Contudo, seu processamento é local, sem distribuição

do comportamento. Além disso, a programação dos *plugins* é feita no próprio ambiente em que o sistema é executado. A proposta apresentada neste artigo amplia o conceito dos *plugins* do Narval para um ambiente SOA. Dessa forma, os *plugins* devem ser serviços web conectados dinamicamente ao SAP, assim como sua execução se torna distribuída, liberando o processamento do SAP para o gerenciamento, coreografia e orquestração das ações em si.

Já o projeto Siri [19] se utiliza de informações de preferências pessoais dos indivíduos e de um histórico de interação para ajudar a resolver tarefas específicas. É baseado na web e links de resultados de buscas. O usuário diz para seu assistente o que quer fazer e este pesquisa em fontes de informações para poder auxiliar o usuário. O software evolui com a experiência de interação. Ele se utiliza também do contexto pessoal (lugar, horário, histórico) do usuário. O Siri foi adquirido pela Apple e é um sistema fechado e proprietário. Entretanto, o mesmo serve de inspiração para o modelo proposto na medida em que o SAP age para buscar na Internet informações que podem servir de auxílio ao usuário, conforme um contexto específico. Outra característica é a forma de comunicação com o usuário. O usuário se comunica e interage com seu SAP como se estivesse conversando com outra pessoa.

Outro projeto bastante relevante é o PAL (*Personalized Assistant that Learns*) [14], financiado pela agência americana DARPA. Este tem por objetivo auxiliar os usuários nos trabalhos com sistemas computacionais e automatizar tarefas rotineiras para liberar as pessoas para tarefas mais importantes [12]. A intenção é criar um sistema cognitivo que possa raciocinar, aprender e lidar com situações imprevistas como forma de fornecer assistência em situações militares, especificamente. Por ser um projeto quase que militar, muitas informações mais aprofundadas são limitadas. Este projeto é conduzido por pesquisadores em inteligência artificial, percepção, aprendizado de máquina, processamento de linguagem natural, representação de conhecimento, diálogo multimodal, ciberconsciência, interação homem-máquina e planejamento flexível [14]. Este projeto supre quase todos os requisitos funcionais desejáveis para SAPs, a não ser a integração com processos de negócio da empresa. Contudo, a proposta apresentada neste artigo busca também a padronização na criação de assistentes pessoais por meio de uma arquitetura de referência, e não especificamente criar comportamentos em si. Os comportamentos são baseados na distribuição desses na forma de *plugins,* serviços web, distribuídos na Internet. Os esforços apresentados no projeto PAL podem e devem servir de inspiração para a criação de comportamentos de planejamento, aprendizado, inteligentes, etc., que podem ser criados por terceiros e utilizados pelo SAP apresentado aqui.

Neste ponto foram apresentados alguns esforços no desenvolvimento de SAPs. Porém, não foi encontrado qualquer modelo ou arquitetura de referência que visasse a criação de SAPs abertos ou com integração aos processos de negócios das empresas. Na maioria dos projetos encontrados, os comportamentos ou são específicos já criados pela própria empresa, ou se em formato plugável (como no projeto Narval), estão em um formato próprio do projeto. Caso se deseje adicionar uma nova funcionalidade, os novos comportamentos devem ser desenvolvidos especificamente para tal projeto e no formato definido por eles, não havendo um padrão realmente aberto que permita uma escalabilidade em larga escala das funcionalidades.

III. PROPOSTA DE ARQUITETURA DE REFERÊNCIA PARA SOFTWARES ASSISTENTES PESSOAIS

Primeiramente, a concepção de uma arquitetura de referência é vista como uma forma de apresentar um padrão genérico para um projeto e deve abordar os requisitos para o desenvolvimento de soluções, guiado pelo modelo de referência e por um estilo arquitetural, de forma a atender as necessidades do projeto [10], e um modelo de referência é uma divisão de funcionalidades, juntamente com o fluxo de dados entre as partes, e possuem características de domínios maduros, decorrente da experiência sobre este domínio. Ele se caracteriza como um padrão de decomposição do problema. Já o estilo arquitetural ou modelo de arquitetura descreve os tipos de elementos e suas relações, juntamente com um conjunto de restrições sobre como eles podem ser utilizados, além de padrões de interação entre os elementos. Tais restrições, sobre a arquitetura e sobre o sistema em si, são vistas na forma de uma imagem da utilização do sistema como um todo [1].

Para o modelo e arquitetura propostos, uma abordagem considerada essencial e que norteia as suas concepções é arquitetura orientada a serviços – SOA (*service oriented architecture*). Este representa uma nova forma de pensar quanto ao projeto da arquitetura de um sistema, quanto à sua filosofia de desenvolvimento, e quanto a sua posterior integração a outros sistemas.

O princípio que rege SOA é de que o desenvolvimento de uma aplicação grande, monolítica, toda nova e complexa deve ser evitada e substituída por um conjunto de aplicações pequenas, já existentes (quando possível) e mais simples [7]. Ou seja, uma aplicação passa a ser fisicamente composta por vários e pequenos módulos de software especializados, distribuídos, acessados remotamente, interoperáveis e reutilizáveis, que são "inteligentemente" unidos conforme o processo, seguindo padrões, podendo a aplicação ser fácil e rapidamente (re)composta para um (novo) processo desejado dado ao baixo acoplamento da aplicação/serviços. A tais módulos dá-se o nome de serviço de software, que pode ser implementado em várias tecnologias, sendo o padrão de facto o de serviços web. Para a visão de SAP deste trabalho, este então pode ser visto como um projeto SOA, em que seus comportamentos (funcionalidades) são modelados como serviços, invocados de repositórios locais da empresa ou de provedores externos de serviços de software. Esta visão dá bases para o SAP para não apenas se adaptar a novos processos como também aos diferentes ambientes computacionais onde ele vier a ser executado. A seleção de serviços mais adequados para o processo pode ser feita dinamicamente (portanto, não definida a priori, rigidamente, e considerando inclusive aspectos de QoS [15]. Graças ao baixo

acoplamento dos serviços (do SAP) e conforme o ambiente da empresa de disponibilização de serviços de software, dos modelos arquiteturais, disponibilização de serviços (sob demanda) e até mesmo de pagamento podem ser adotados [9].

Sob a ótica dessas abordagens de base, quatro aspectos principais balizaram o desdobramento dos requisitos gerais de SAP no modelo e arquitetura de referência:

- SAPs devem ser concebidos para permitir sua integração aos processos de negócios das empresas, como mais uma "peça" de um grande ambiente de operações da empresa. Com isso ele deixa de ser um sistema puramente reativo à interação do usuário, mas também proativo, executando tarefas de processos de negócios por ele.
- SAPs devem ser tanto flexíveis para se adequar aos processos e transações empresariais. como sua arquitetura funcional escalável para poder incorporar novas funcionalidades e processos de negócio. Cada processo de negócio tem inúmeras particularidades, tanto de contexto como de ambientes computacionais envolvidos. Portanto, há necessidade de que se adaptem ao ambiente geral de execução. Isso é potencializado pela noção de serviços de software, em que SAPs devem poder ter novas funcionalidades ou mesmo ter a possibilidade de "escolherem" de onde deverão acessar a funcionalidade requerida ou desejada.
- SAPs devem ser concebidos para interoperar. Isso significa que devem usar tanto quanto possível especificações ou padrões abertos de projeto, comunicação e integração, criando-se um cenário aberto.

A. Modelo de Referência

Entre os elementos principais do modelo de referência para assistentes pessoais proposto nesse trabalho (Fig. 1) está a **Ação**, que se refere aos comportamentos do assistente, a **Interação** para com o usuário ou outros assistentes ou softwares, e o **Gerenciamento**, para gerenciar a execução do assistente, organizar as informações e o fluxo dessas entre os demais elementos.

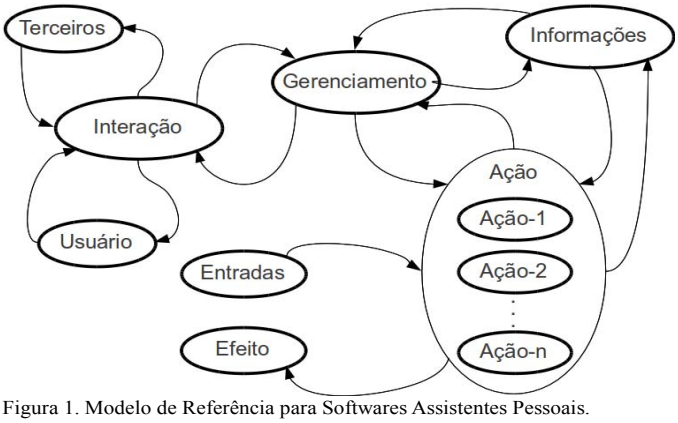

Figura 1. Modelo de Referência para Softwares Assistentes Pessoais.

A **Ação** é a forma como o assistente age, que é a composição do conjunto de *n* ações que podem ser utilizadas pelo SAP. As Informações são os elementos úteis para a execução de uma determinada ação ou de um conjunto de ações. As Entradas referem-se ao conjunto de informações que chegam de elementos externos ao assistente e que servem para iniciar uma determinada ação do assistente. Essa ação reage com um Efeito, que pode ser tanto interno do próprio assistente, como externo, com parceiros, outros assistentes, outros softwares, ou mesmo pela interação com o usuário.

O **Gerenciamento** é responsável pela organização das informações do usuário, pela execução das ações e pelo fluxo dessas informações entre os elementos do modelo. Um conjunto de informações acerca do usuário do SAP e das ações é necessário para que o assistente gerencie sua execução. Elas devem evoluir com o tempo e em concordância com a evolução do usuário, tarefas, preferências, etc. Por meio delas, o Gerenciamento deve decidir quando e como iniciar uma ação, ou seja, tornar uma ação operacional quando certas condições forem satisfeitas.

A **Interação** é a forma como se dão as comunicações e negociações entre o usuário e módulo de Gerenciamento do SAP, ou entre o módulo de Gerenciamento e outros elementos.

B. Arquitetura de Referência

A existência de uma arquitetura de referência para assistentes pessoais é relevante, mas não suficiente para fazer frente àqueles requisitos anteriormente expostos. É importante que o assistente possa ter a característica de comportamentos "plugáveis", se manter em um padrão de comunicação e ainda trabalhar com a questão de serviços de software, propiciando assim as bases para a flexibilidade e escalabilidade desejadas. Assim, os comportamentos do assistente pessoal são habilitados através da invocação de serviços web, distribuídos na Internet, na forma de uma "federação" de serviços [22], também podendo ser vista como uma "nuvem" dentro da qual estão inseridos os vários repositórios de serviços, fornecidos por diferentes empresas, organizações ou desenvolvedores.

Com base no modelo de referência anteriormente exposto, a Fig. 2 apresenta a arquitetura de referência para SAP proposta. Ela corresponde a uma derivação do modelo abstrato, e é independente de modelo e de tecnologia de implementações.

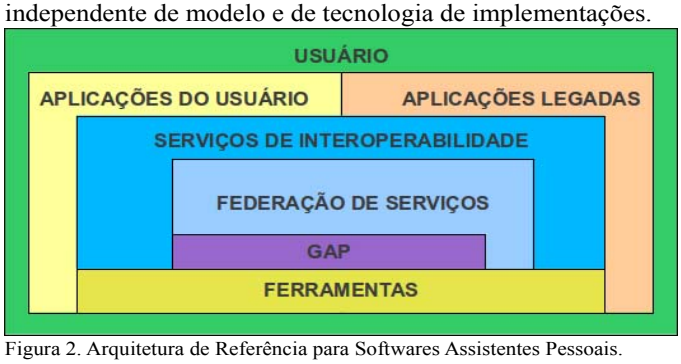

Figura 2. Arquitetura de Referência para Softwares Assistentes Pessoais.

- **Aplicações Legadas:** Softwares desenvolvidos pela empresa, ou de terceiros, para serem integrados ao SAP.
- **Aplicações do Usuário:** Aplicações que podem servir para interação com o SAP. No projeto Sandy [11], por exemplo, é feito por serviços de e-mails, mensagens via Twitter, sistemas de IM, ou mesmo SMS.
- **Ferramentas:** Aplicações de configuração de SAPs, de comportamentos ou ambientes de desenvolvimento.

A Fig. 2 mostra também uma visão de como o usuário está em contato com o SAP. O usuário pode estar em vários níveis de conhecimento:

- **Usuário Simples:** Pode cadastrar um assistente pessoal, criar e configurar comportamentos e as informações dos comportamentos do seu assistente.
- **Usuário Avançado:** Pode compor comportamentos mais complexos no assistente por meio da composição de condicionais, laços e orquestração de serviços web via uma interface de algoritmos ou fluxogramas, textual ou preferencialmente gráfica.
- **Desenvolvedor de serviços web:** Desenvolve serviços web que podem ser utilizados para compor comportamentos.
- **Desenvolvedor de serviços de interoperabilidade:** Desenvolve interfaces de serviços web para sistemas desenvolvidos sem essa tecnologia (e padrões associados).
- **Desenvolvedor de Assistentes Pessoais:** Desenvolve núcleos de execução de SAPs e interfaces de configuração.

A área mais externa não é exatamente um elemento da arquitetura, mas sim a visão que o usuário possui do assistente pessoal, ou seja, os elementos da arquitetura da qual devem fazer o interfaceamento entre o usuário e seu SAP .

A Federação de Serviços representa o conjunto de serviços web distribuídos na Internet e que podem ser utilizados nos comportamento do SAP isoladamente, ou podem servir para compor outros serviços web (orquestração e composição de serviços).

Os Serviços de Interoperabilidade, por sua vez, são criados para servirem de interface com outros sistemas que não estão no padrão SOA de serviços web. Neste caso, adota-se uma abordagem de integração onde os aspectos de interoperabilidade são tratados de forma desacoplada dos aspectos intrínsecos funcionais e não funcionais de um serviço, como proposto no projeto COIN [5].

O Gerenciador de Assistentes Pessoais (GAP) é responsável pela coordenação da execução dos comportamentos e suas informações. Os comportamentos podem ter três níveis básicos. Simplificadamente:

- **Comportamento Simples:** Composto de informações para a chamada do serviço web. Este é executado toda vez que o comportamento é chamado, enviando os parâmetros de entrada e retornando o resultado.
- **Comportamento Condicional:** Modela comportamentos com especificações condicionais (regras *if-then*).
- **Comportamento Complexo:** O usuário, com conhecimento mais avançado, pode modelar o comportamento na forma de um fluxograma, ou algoritmo.

Os comportamentos são modelados em uma interface de configuração do SAP, sendo que a forma como isso é feito depende de cada desenvolvedor. Porém, partindo-se de uma arquitetura de referência, garante-se que instâncias de SAPs devem ser coerentes com o modelo maior, mesmo que tecnologias mudem ou que funcionalidades num dado momento não presentes no SAP venham a ser introduzidas no futuro.

IV. VERIFICAÇÃO

Uma das formas de verificação da Arquitetura de Referência é a factibilidade de criação de uma instância implementada dela. Para tal, foi gerado um protótipo baseado em um estudo de caso específico, ou seja, uma instância que implementa instâncias de comportamentos de um SAP.

Neste estudo de caso, é considerado o cenário em que um funcionário de uma empresa tem a função de atuar em cima do processo de negócio gerenciamento do estoque de produtos. De acordo com a lógica desse processo, quando o estoque de um produto passa de um limite mínimo, fica a cargo desse funcionário efetuar nova compra, de modo a manter o estoque sempre em dia e não permitir que a produção fique parada por falta de matéria-prima. Conforme produtos são retirados do estoque, sua quantidade diminui. No final do dia, o funcionário faz um relatório das atividades realizadas durante todo o dia, desde o início do processo de uma compra, até as modificações em uma ordem de compra e fechamento da ordem. O funcionário possui acesso a um sistema legado de controle de estoque, com seu próprio banco de dados, por meio de um login de usuário e uma senha. Neste caso, o SAP deve ser utilizado para auxiliar o funcionário a gerenciar o sistema de controle de estoque e para a criação do relatório das atividades diárias. Ainda, é considerado que o usuário deseja se comunicar com o seu SAP por meio de aplicações que não precisam ser instaladas em seu computador e que podem ser acessadas de qualquer lugar, inclusive do seu dispositivo móvel. No caso instanciado implementado, o usuário pretende se comunicar com seu SAP como se fosse uma outra pessoa, por meio de mensagens via Twitter, Gtalk, e-mail ou mensagens de texto no celular.

A. *Protótipo*

Sob o cenário apresentado alguns elementos de implementação para compor o SAP para este estudo de caso foram derivados a partir da Arquitetura de Referência proposta (Fig. 3).

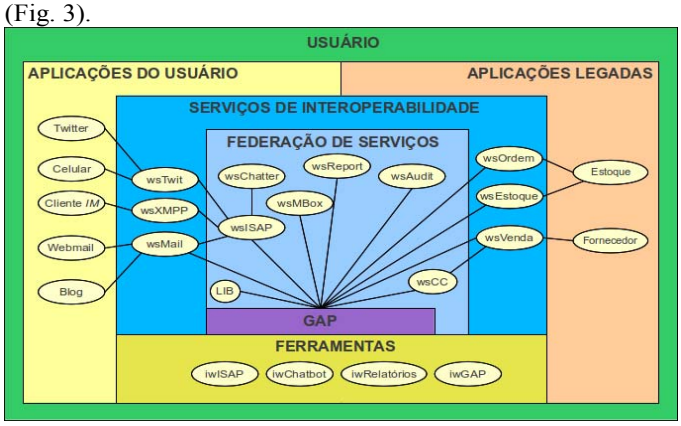

Figura 3. Arquitetura derivada para o caso particular.

Considerando a visão SOA do modelo, a arquitetura de referência pode ser instanciada usando diferentes abordagens, com diferentes serviços (funcionalidades), ferramentas de implementação e mecanismos de comunicação. Mais do que isso, o SAP passa a ser não um software monolítico, mas sim

um sistema distribuído, aberto, que se comunica via rede, envoltos num mesmo arcabouço lógico.

As implementações feitas são aqui apresentadas conforme sua classificação na Arquitetura de Referência proposta. Todos os sistemas *daemons* foram desenvolvidos em linguagem de programação Java, os serviços web em linguagem PHP (no estilo SOA) e as interfaces em PHP e HTML via web.

Aplicações Legadas:

- Estoque: Sistema de controle de estoque com informações de produtos, fornecedores, quantidades em estoque, a serem utilizadas para iniciar processos de compra.
- Fornecedor: Controle de produtos no fornecedor. Informa os produtos, quantidades, e preço.

Aplicações do Usuário:

- Twitter: Serviço de publicação de mensagens (*microblog*) de até 140 caracteres.
- Celular: Via Twitter, pode-se habilitar receber/enviar mensagens via mensagem de texto no celular.
- Cliente IM: Para troca de mensagens com o SAP. O usuário vê seu assistente pessoal como uma pessoa de seus contatos conectada ao serviço.
- Webmail: Para troca de mensagens de e-mail com o SAP.
- Blog: O SAP envia postagens a um blog para publicar relatórios relatórios públicos.

Os **Serviços de Interoperabilidade**, por sua vez, são serviços que fornecem uma interface de serviço web SOA da OASIS para outros sistemas que não possuem tal interface:

- wsEstoque: Acessa informações do sistema de estoque.
- wsOrdem: Gerenciamento das ordens de compra.
- wsVenda: Acessa o sistema automático de vendas, um *daemon* executando do lado do sistema de fornecedores.
- wsTwit: Acessa uma conta no Twitter.
- wsXMPP: Acessa um *daemon* de conexão ao IM Gtalk.
- wsMail: Acessa os e-mails de um usuário em um servidor.

Federação de Serviços.

- wsCC: Simula transações com cartão de crédito.
- wsAudit: Armazena informações de auditoria.
- wsReport: Geração automática de relatórios. Recebe mensagens referente as atividades e quando requisitado retorna o relatório compilado.
- wsMBox: Gerencia mensagens provindas do ISAP.
- wsChatter: Gera respostas para o usuário e quem mais se comunicar com o assistente pessoal tal como um *chatbot.*
- wsISAP: Serviço de interface com o usuário. Se utiliza de serviços como o wsChatter, wsXMPP e wsMail para trocar mensagens com o usuário. Há um sistema *daemon* que faz o gerenciamento das mensagens. Quando um usuário se comunica com a Interface Social para Assistentes Pessoais (ISAP) a mensagem é avaliada e enviada ao GAP, decidindo a melhor forma de resposta.
- LIBS: Bibliotecas utilizadas para criar os comportamentos (string, math, datetime, kqml, myMessageTable).
- Ferramentas: Programas de interfaces web para configuração de alguns sistemas.
- iwISAP: Interface para configuração do ISAP do assistente pessoal.
- iwChatbot: Configuração *chatbot* do SAP. Para *feedback* das requisições do usuário.
- iwRelatorios: Configuração do gerador de relatórios.
- iwGAP: Ambiente de configuração do assistente pessoal e seus comportamentos. Os comportamentos são aqui configurados na forma de um algoritmo. O GAP (Gerenciador de Assistentes Pessoais) é um *daemon* que gerencia a execução autônoma dos assistentes pessoais.

Havendo esse ambiente aberto de softwares e serviços, foi criado um assistente pessoal e alguns comportamentos para efetuar o gerenciamento do sistema de controle de estoque, de responsabilidade do usuário. Em verdade, neste protótipo existem dois caminhos distintos para a criação do comportamento automático de compra:

1. **Usuário avançado:** pode criar comportamentos na forma de algoritmos na interface de configuração do SAP;
2. **Usuário básico:** Requisita a um desenvolvedor da empresa um serviço web que efetua o processo requerido. O usuário utiliza este serviço web no seu comportamento;

Aqui, neste estudo de caso, é considerado que o usuário é do tipo avançado e configurou as informações e algoritmos que efetua a compra automática. Os comportamentos desenvolvidos são apresentados a seguir:

Neste caso, os seguintes comportamentos foram criados:

- AtualizaHora: Invoca o serviço datetime para atualizar data e horário, a serem utilizadas pelos comportamentos.
- ISAPalive: Envia, a cada pouco, uma mensagem ao ISAP, informando que ele deve se manter on-line.
- mbox: Invoca operações do wsISAP para gerenciar mensagens do usuário para o GAP.
- Report: Em um horário configurado pelo usuário, gera relatórios das últimas atividades. Relatórios públicos são enviados ao Blog e privados são enviados por e-mail.
- Compra-ordem: Se um produto está com estoque baixo, gera uma ordem de compra. Essa ordem é enviada ao fornecedor escolhido pelo sistema gerenciador de estoques. Cada nova ordem criada é enviada uma mensagem ao usuário e uma atividade ao wsRelatorio.
- Compra: Se há uma ordem aberta, verifica o estado dela no fornecedor. Se esta foi alterada, altera-a também no wsOrdem e envia uma mensagem ao usuário do assistente pessoal, requisitando confirmação. Se a ordem foi aceita pelo usuário, efetua o pagamento, finalizando a compra, se cancelada, seleciona outro fornecedor.
- Compra-altera: Aguarda a confirmação do usuário de uma ordem alterada. Se cancelada, então cancela a ordem no wsOrdem e no wsVenda.

O GAP implementado executa os comportamentos por ciclos. Cada comportamento pode conter diversas atividades e em cada ciclo uma atividade de cada comportamento é executada. Na Fig. 4 é possível visualizar o editor de comportamentos do assistente pessoal desenvolvido.

A esquerda da interface há uma área para configurar as informações básicas do comportamento, ligar/desligar sua execução, efetuar um *reload* do comportamento e lista de links de repositórios de provedores de serviços web.

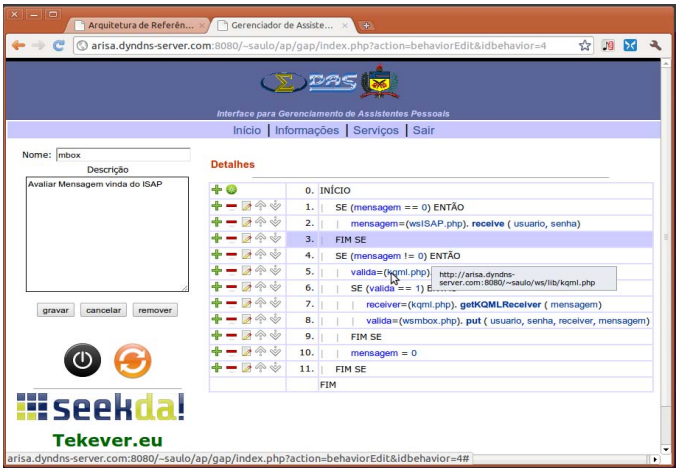

Figura 4. Editor de comportamentos do Assistente Pessoal.

A direita da Fig. 4 há o editor de comportamentos propriamente ditos, com opções de criação de condicionais, laços, atribuição de valores a variáveis e configurações de chamadas de serviços web.

Para que o SAP possa ser visto pelo usuário como se fosse outra pessoa conectada na internet e interagindo com ele, é necessário a criação de contas nos tipos de serviços utilizados pelo ISAP, tal como conta de e-mail (no caso de ser criada uma conta no Gmail, a mesma já pode ser utilizada pelo assistente pessoal como conta no Gtalk), conta de Gtalk, conta no Twitter e um Blog). Nessa implementação foi criada uma assistente pessoal de nome Arisa (Acrônimo para *Assistant Representative: an Instance using Services Architecture*), com suas respectivas contas para serem utilizadas pelo ISAP: (i) Gmail e Gtalk: *personal.assistant.arisa@gmail.com*; (ii) Twitter: *@arisa_ap*; e (iii) Blog: *http://www.projetoarisa.com.br*.

Também foram cadastradas informações de dois fornecedores de produtos e informações referentes ao sistema de estoque que o SAP deve gerenciar.

B. Teste

No sistema de estoque, unidades do produto mouse (código 001) são vendidas, ficando suas informações da seguinte forma: quantidade = 3, quantidade mínima = 10 e quantidade máxima = 20. Ou seja, quando a quantidade deste produto estiver abaixo de 10, são comprados os produtos necessários para chegar à quantidade máxima. Este produto é fornecido pelos fornecedores, de nomes fictícios, *John Doe Ltd*a e *Sora Konpyuuta* ao valor de 25 reais e 28 reais a unidade, respectivamente.

No caso, o primeiro fornecedor possui apenas 10 produtos em estoque para venda, e o segundo possui 50. O fornecedor selecionado inicialmente para a compra é o primeiro que foi cadastrado no sistema como fornecedor, no caso foi selecionado o fornecedor *John Doe Ltda*. As trocas de mensagens entre o SAP e usuário para efetuar o gerenciamento das ordens de compra são visualizadas na Fig. 5.

Quando o assistente pessoal verifica que o estoque está baixo, é iniciado o processo de compra. Como a quantidade do produto 001, do fornecedor *John Doe Ltda*, não é o suficiente para o total da ordem de comprar (i.e. 10 produtos), então a ordem é modificada e o assistente requisita confirmação da alteração. Quando o usuário necessitar comprar todos os produtos para repor o estoque no máximo, a ordem deve ser cancelada e uma nova criada, para outro fornecedor. Tendo corrido tudo corretamente com a segunda ordem criada, o pagamento é efetuado e a ordem é fechada, atualizando os estoques no fornecedor e no cliente. Por fim, um relatório completo é enviado por e-mail no horário definido pelo usuário.

Um ponto a ser observado é que este teste foi realizado com o usuário conectado em seu usuário no Gtalk. Caso o usuário não estivesse conectado, as mensagens seriam trocadas via mensagem de texto do celular.

Figura 5. Troca de mensagens entre o assistente pessoal e o usuário.

Em tempo, é importante observar que pode não ser interessante que o usuário fique recebendo informações do SAP a todo o momento, ou mesmo o usuário pode querer que esse cancelamento de ordem de compra seja automático. Contudo, neste exemplo várias mensagens são trocadas com o usuário para fins de testes e verificação do funcionamento do comportamento.

V. CONSIDERAÇÕES FINAIS

Este artigo apresentou uma proposta para uma Arquitetura de Referência para Softwares Assistentes Pessoais. Essa proposta se apropriou da Arquitetura Orientada a Serviços para servir de estilo arquitetural.

Com base nos resultados de testes sob a implementação final, verificou-se que esta instância se comportou conforme o que foi proposto na arquitetura de referência e executou corretamente as ações associadas aos comportamentos do processo de negócio envolvido no estudo de caso. Embora apresentado apenas parcialmente neste artigo, o SAP proposto foi implementado em sua totalidade. Quanto ao desempenho, embora neste momento do trabalho não tenha sido o principal foco, apresentou boa estabilidade perante as atividades realizadas em uma arquitetura distribuída, mas em rede local.

A instância derivada mostrou-se coerente com o modelo e arquitetura de referência, o que garantiu a coerência do SAP derivado. Com a implementação realizada, mostrou-se

igualmente que é possível realizar aquela derivação assim como ter um SAP integrado ao ambiente de negócios da empresa, interoperando com vários subsistemas, dentro de uma arquitetura aberta e de intenso uso de padrões.

A proposta aqui apresentada, além de apresentar uma inovadora arquitetura de referencia para a criação de qualquer tipo de SAP, fornece indicativos da criação de comportamentos baseados em SOA. Este estilo arquitetural aberto permite que funcionalidades do SAP não precisem ter sido criadas especificamente para o SAP do usuário em si, mas podendo ser utilizadas para a composição de comportamentos de forma dinâmica. Isso também permite que a própria empresa do usuário forneça funcionalidades que façam a integração do SAP do usuário com os processos de negócio da empresa. Outra vantagem é que funcionalidades podem ser desenvolvidas por terceiros, e utilizadas pelo usuário de forma gratuita ou paga. Isso pode levar a um foco específico de negócios, que é a criação de comportamentos e funcionalidades específicas para SAPs, apenas criando serviços web e disponibilizando-os aos usuários para a composição dos seus comportamentos. Como os serviços web vêm se firmando como um padrão de mercado, isso provê, consequentemente, opções de funcionalidades e comunicação para quem queira desenvolver SAPs que sigam a proposta.

Contudo, ainda se constata que a área de SAPs tem ainda enormes desafios, incluindo ao que se refere a limitações das TIs atuais para certos propósitos (por exemplo, o problema da interoperabilidade semântica), os impactos organizacionais, a complexidade do problema de composição de comportamentos dentro da abordagem SOA, etc.

Neste sentido, este trabalho deve ser visto como uma contribuição para a área de SAPs, indo na direção dos inúmeros requisitos, em vários níveis, identificados na literatura. Existem ainda vários aspectos que devem ser mais bem estudados e implementados. Os próximos passos de curto prazo focam no aprimoramento da questão de composição de serviços e da "inteligência" quando as tomadas de decisão, ou seja, em relação à autonomia do assistente.

VI. AGRADECIMENTOS

Este trabalho foi parcialmente financiado pela CAPES e CNPq.

REFERÊNCIAS

[1] Bass, Len; Clements,Paul and Kazman, Rick. "Software architecture in practice". Addison-Wesley, 2003.

[2] Bocionek, S. "Software secretaries: learning and negotiating personal assistants for the daily office work In Systems, Man, and Cybernetics", In: Humans, Information and Technology. 1994 IEEE International Conference on, 12 vol.1. 2-5 Oct. 1994.

[3] Bush, J.; Irvine, J. and Dunlop, J. Personal Assistant Agent and Content Manager for Ubiquitous Services. Wireless Communication Systems, 2006. ISWCS'06. 3rd International Symposium on , 169-173. 2006.

[4] Chauvat, N. "Narval, the intelligent personal assistant or how the french Linux Gazette is built", In: Linux Gazette, issue 59. Nov, 2000. Disponível em <http://linuxgazette.net/issue59/chauvat.html>, acessado em 14/02/2011.

[5] COIN. "Coin: enterprise collaboration & interoperability", Disponível em: <http://www.coin-ip.eu>. 2009. Acessado em Jun/2011.

[6] ebXML, Electronic Business using eXtensible Markup Language, Disponível em <http://www.ebxml.org/> Acessado em Jun/2011.

[7] Huhns, M.N.; Singh, M.P., "Personal assistants", In: IEEE Internet Computing, Vol. 2, Issue 5, pp. 90-92. Sep./Oct. 1998.

[8] Huhns, M.N. "Agents as Web services", In: Internet Computing, IEEE 6, 93-95. 2002.

[9] IBM. Software as a service: Build a Web-delivered SaaS framework for forms and workflow-driven applications. <http://www.ibm.com/developerworks/architecture/library/ar-saasframe/>. 2008. Acessado em Jun/2011.

[10] MacKenzie, C.; Laskey, K.; McCabe, F. at all. "Reference Model for Service Oriented Architecture 1.0", In: OASIS Standard, 12 October 2006. <http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf> acessado em Fev/2009.

[11] Mann, H. "Free Personal Assistant - Meet Sandy". Disponível em <http://honestholly.com/free-personal-assistant-meet-sandy/>, acessado 14/02/2011.

[12] Markoff, J. "A Software Secretary That Takes Charge", In: New York Times, 13/12/2008. Disponível em: <http://www.nytimes.com/2008/12/14/business/14stream.html?_r=1&scp=7&sq=personal%20assistant&st=cse>. Acessado em: 17/03/2009.

[13] Michaell, T.; et all. "Experience With a Learning Personal Assistant", In: Communications of the ACM, July, 1994.

[14] PAL, PAL Project, <http://pal.sri.com/>, acessado em 14/02/2011.

[15] Perin de Souza, A.; Rabelo, R. J. "Uma solução aberta e flexível de descoberta de serviços para maior agilidade na integração BPM&SOA". Simpósio Brasileiro de Sistemas de Informação / V Workshop de Gestão de Processos de Negócios, p. 509-516, 2011.

[16] RosettaNet, Disponível em <http://www.rosettanet.org/>. Acessado em Junho/2011.

[17] SHELLTOYS. "Personal assistant - day planner and personal information manager". Disponível em: <http://www.shelltoys.com/personal_assistant/>, acessado em 14/02/2011.

[18] Schiafino, S., Amandi A.; Polite Personal Agents. in IEEE Intelligent Systems, p12-18. Jan-Fev 2006.

[19] SIRI, disponível em <http://siri.com/>, acessado em 14/02/2011.

[20] Thenault, Sylvain. Narval-moved. Logilab Project. Disponível em: <http://www.logilab.org/908/>, Acessado em 14/02/2011.

[21] UBL, Disponível em <Universal Business Language, http://www.oasis-open.org/committees/ubl/> Acessado em Jun/2011.

[22] Zambiasi, S.P.; Rabelo R.J.. "Uma arquitetura de referência para softwares assistentes pessoais baseada na arquitetura orientada à serviços", In: I2TS'2010: 9th International Information and Telecommunication Technologies Symposium, 2010.

Saulo Popov Zambiasi possui graduação em Ciência da Computação pela Universidade do Oeste de Santa Catarina - Campus de Chapecó (1998), especialização em Ciência da Computação pela Universidade Federal de Santa Catarina (2000) e mestrado em Ciências da Computação pela Universidade Federal de Santa Catarina (2002). Atualmente é Pesquisador - Estudante do Programa de Pós-Graduação em Engenharia de Automação e Sistemas da Universidade Federal de Santa Catarina. É professor da Universidade do Sul de Santa Catarina (UNISUL), das Faculdades Barddal e Revisor de periódico da Revista IEEE América Latina. Tem experiência na área de Ciência da Computação, com ênfase em Computação Aplicada. Atuando principalmente nos seguintes temas: Inteligência Artificial Distribuída, Sistemas Multiagentes, Agentes Inteligentes, Computação Gráfica, Jogos de Computador e Automação Residencial.

Dr. Ricardo José Rabelo concluiu o doutorado em Engenharia Elétrica pela Universidade Nova de Lisboa (UNL) em 1997. É Professor Associado do Departamento de Automação e Sistemas da Universidade Federal de Santa Catarina (UFSC) desde 2000, é coordenador do GSIGMA Grupo de Sistemas Inteligentes de Manufatura. Suas atuais áreas de interesse incluem: redes colaborativas organizações, arquitetura orientada a serviços, gestão de conhecimento, sistemas multiagentes, e sistemas de apoio à decisão multi-critérios. Esteve envolvido em vários projetos de pesquisa europeus e brasileiros, tem mais de 90 publicações, incluindo conferências, periódicos e capítulos em livros. É também membro de diversas comissões do programa conferências nacionais e internacionais.